National College of Ireland

A novel model for data storage using LZW compression technique for Cloud based Electronic Healthcare Systems

MSc Research Project
Cloud Computing

Srija Perugu
Student ID: X21168105

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Srija Perugu
Student ID:	X21168105
Programme:	Cloud Computing **Year:** 2022-2023
Module:	Msc Research Project
Supervisor:	Vikas Sahni
Submission Due Date:	15-12-2022
Project Title:	A novel model for data storage using LZW compression technique for Cloud based Electronic Healthcare Systems
Word Count:	6585 **Page Count** 20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: P.Srija

Date: 15-12-2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A novel model for data storage using LZW compression technique for Cloud based Electronic Healthcare Systems

Srija Perugu
X21168105

Abstract

In recent years, there has been a significant shift toward storing electronic health records in mobile cloud environments, wherein mobile devices are integrated to cloud computing to facilitate medical information exchange between healthcare providers and patients. This cutting-edge model has made it possible to provide healthcare services due to the adaptability and efficacy, accessibility of electronic health records (EHRs). E-health systems, which store and transfer patient information online, have their privacy and security compromised by this new paradigm.

A new model for sharing EHRs is developed using the blockchain and distributed interplanetary file system (IPFS) on a mobile cloud platform. To achieve truly decentralised data storage and sharing, blockchain has been integrated into a distributed interplanetary file system. In this model, all patient medical records are stored on a server running a distributed interplanetary file system. Since this server has limited memory, it becomes a problem if all files are stored in plain text format. This paper present a model here that makes use of the LZW compression method to first compress all of the medical records and then store the resulting compressed file in order to cut down on storage space requirements.

1 Introduction

Blockchain distributed as well as trustworthy nature has shown great promise in several areas of e-health, including the safe exchange of Electronic Health Records as well as the administration of data access by a variety of medical organisations. Therefore, blockchain technology has the potential to fundamentally change the healthcare industry by bringing about numerous improvements in the quality of care provided to patients. (A. Dubovitskaya, 2017) (S. Jiang, 2018).

In a blockchain-based healthcare system, patients would have access to and ownership of their own medical records, which could increase the security and trustworthiness of that data. The safekeeping of patients' medical records is crucial. Because of the importance and sensitivity of these records, they are frequently the target of cyber-attacks. All private information must be kept safe (M. Hölbl, 470).

There is also the matter of data management, which should ideally fall under the patient's scope. Accordingly, another application where cutting-edge technologies can be useful is in facilitating the sharing and accessing of the control of patients' healthcare data. Blockchain technology is highly resilient to failures and attacks, and it also offers a variety of access control mechanisms. Therefore, blockchain is a useful infrastructure (M. Hölbl, 470).

It is inefficient to store huge files on the blockchain. Due to the restrictions imposed by the size of individual blocks, files must be broken up and reassembled outside of the blockchain. There is a need to store extra information for the purpose of file reconstruction, which would necessitate either more storage space or a separate system to supply the reconstruction details. To facilitate data access and keep track of reassembly instructions, smart contracts could be used to directly store file parts. However, executing smart contracts on every mining as well as verifying node makes sending as well as storing large files, also partially, costly (for example, in terms of gas costs) (Steichen M. F., 2018).

InterPlanetary File System is an intriguing option because it combines the benefits of file sharing with those of the aforementioned hashes. IPFS uses cryptographic hashes to verify the integrity of files and to transfer them. Files kept on InterPlanetary File System can be requested as well as viewed by anybody who has access to or has set up an InterPlanetary File System node, much like public blockchain. Large files containing private or sensitive information present a challenge for blockchain applications (Benet, 2014).

The term "coding" is often used to describe the process of data compression, with the implication that "coding" refers to any special representation of the data that meets a specific requirement. The study of secure encoding is what information theory is all about. A subfield of information theory, data compression seeks to reduce the total volume of data that must be transmitted. Compressing data serves a crucial function in both transmission and storage. Data compression is the process of reducing the number of times a piece of information is stored by eliminating unnecessary details. It made less use of things like storage space and bandwidth. Two main types of data compression are lossless and lossy. It's common practise to compress text without losing quality, but images are typically compressed using lossy methods (Kaur, 2012).

Huffman and LempelZiv-Welch (LZW) arithmetic coding are just two examples of the many algorithms used for data compression. The LZW algorithm is most commonly used compression method. A greedy approach is similar to the LZW algorithm, which splits the text into smaller strings. When using LZW compression, strings of text are exchanged for numerical representations. The LZW algorithm is a dictionary-based lossless data compression algorithm. This LZW compressor keeps track of character records from the input file. In the dictionary, the individual characters are represented by index numbers (Kaur, 2012).

Research Question

RQ1: To what extent blockchain technology can make electronic health record systems more secure and prevent malicious attack?

Blockchain technology is used by the majority of published works for the purposes of information exchange, medical record keeping, and user authentication. In addition audit trail management, supply chain management and other scenarios including drug prescription management as well as auditing are rare uses of blockchain. To address blockchain limitations in storing massive amounts of data, a popular system now combines the blockchain with the InterPlanetary File System. The network's overall speed decreases as a result of the delay in replicating and storing data due to its bulk. To remedy this, IPFS storage was implemented for archiving the information and transmitting the immutable URLs to the chain.

RQ2: To what extent is an IPFS useful in healthcare and how did it change over time?

When a block chain is integrated with an IPFS server, the data can be stored securely. The blockchain will eventually store the hashcode that was obtained via IPFS. If all files are kept in plain text format, the IPFS server will quickly run out of memory. To keep against running out of storage space, the data will be compressed and keep that compacted copy around. Data compression techniques allow for this by compressing the size for text information while storing the same amount of data in fewer bits, which in turn reduces, capacity utilisation, data storage capacity and transmission capacity. Large amounts of computation are often needed due to the enormous quantity of data that needs to be processed.

Contributions

This paper presents a model that utilises LZW compression for EHRs sharing architecture that makes use of blockchain and decentralised Interplanetary File System (IPFS) on a mobile cloud platform to address the problem of memory exhaustion.

This paper also present usability testing on an Android mobile app and Amazon Web Services cloud computing to examine how well the suggested EHRs sharing model works in practise.

2 Related Work

There have been a number of traditional solutions to the problem of sharing secure EHRs in cloud systems. The research (R. Wu, Oct. 2012) focuses on access control challenges in cloud-based electronic medical record systems. Where a systematised access control mechanism supports selective sharing of electronic health records gathered from multiple healthcare providers in the cloud. In addition, they only examined their concept of sharing on such a computer simulation involving Virtual Machines (VMs), ignoring implementation of resource-constrained devices such as smartphones. Public Key Infrastructure has been employed to ensure authentication between engaging healthcare providers as well as the EHR

sharing cloud (A. Ibrahim, May 2016). Further authors in (Z. Ying, 2018) also used CP-ABE, a promising cryptography prototype that can perform both fine-grained access control method and an encryption. In CP-ABE, an access policy is appended to a ciphertext; however, this access policy is not encrypted, leading to privacy leakage.

2.1 LZW Algorithm

The storage overhead caused by the expansion of blockchain network is one of the major limitations. Compression-based strategies generate compressed blocks as well as data which accumulate over time but might not provide sufficient storage on peers. This can be mitigated by modelling compression techniques which provide efficient data representation for IIoT systems, resulting in higher compression ratios (Akrasi-Mensah, 2022).

As a result of the continuous transactions through blockchain, the blockchain data will continue to grow, posing significant challenges for data storage and use. The work in (Du, 2020) proposes a lossless data compression method for text compression based upon LZW (Lemple-Ziv-Welch) algorithm, along with a compression storage as well as sharing framework for medical data utilising blockchain that provide efficient access and secure services for EHRs.

2.2 Blockchain Technology

In the perspective of blockchain technology, numerous studies have investigated blockchain capacity to facilitate the sharing of e-health data. Blockchain was utilised (V. Ramani, 2018) to ensure medical user's access to trustworthy EHRs. The even include on intelligent contracts to manage doctors' EHR usage. Due to the authors' emphasis on theoretical analysis, the viability of proposed solution in actual EHRs sharing scenarios has not been confirmed. Consequently, important aspects of EHRs sharing, such as adaptability, accessibility, and identity management, had not been studied. In the meantime, a blockchain-based data management concept was introduced in (N. Rifi, 2017) to facilitate the secure exchange of EHRs between medical consumers. The work proposed in (Q. I. Xia, 2017) is a system, MeDShare, which addresses these issues of medical data sharing between medical big data custodian in such a trustless environment. The blockchain-based system provides data provenance, auditing, as well as control for shared medical data stored in cloud repositories by big data entities. It monitors entities which access information for malicious purposes on a system belonging to a data custodian. The performance is comparable to that of current cutting-edge solutions for cloud service provider data sharing. By trying to implement MeDShare and cloud service providers as well as other data guardians will indeed be able to perform data provenance as well as auditing whereas sharing medical data which includes entities like research and medical institution with minimal privacy risk.

A smart contract can be said as piece of software that encodes the terms of an agreement between parties who don't necessarily trust one another and then automatically puts those terms into effect. As part of blockchain transaction, a smart contract is either deployed as

well as executed on the network. The blockchain network relies on miners, a special category of participants, to deploy new contracts and carry out existing ones. The computational costs associated with carrying out the contracts are what miners are compensated for. Ethereum as well as Hyperledger Fabric are the most well-known platforms that allow for the deployment and execution of smart contracts (Alharby, 2018).

The authors of (X. Liang, 2017) presented an innovative user-centric method for sharing health data that could improve identity management and protect patient data privacy. To protect data storage, they never had considered the issue of identity management as well as user authentication.

2.3 Interplanetary File System (IPFS)

The authors (S. Wang, 2018) designed a decentralised storage system that incorporates Ethereum blockchain, Interplanetary File System, as well as attribute-based encryption (ABE) in order to better preserve data privacy as well as availability on clouds. In this architecture, no trusted PKGs are required. The data owner can gain fine-grained control over data access by encrypting their data according to a predetermined access policy and then providing secret keys to users. During the same time, focusing upon the smart contract upon the Ethereum blockchain, as well as the keyword search function within ciphertext of a decentralised storage system will be implemented, and also the typical cloud storage problem of the cloud server not returning results or returning incorrect results is resolved.

To solve the issue of storing large amounts of data on blockchain, work of (M. Steichen, 2018) developed a modified version of the Interplanetary File System (IPFS) that integrates Ethereum smart contracts to allow for secure file sharing. Their access control list is maintained by the smart contract, while it is enforced by the modified IPFS software. When a file is downloaded, uploaded or transferred, it interacts with smart contract for this purpose.

Considering data availability and reliability, storage overhead, and other concerns for service providers, the work of(Y. Chen, 2017) designed a zigzag-based storage model to enhance the IPFS block storage model. In addition, blockchain has been utilised to integrate IPFS with the storage model. To solve the issue of storing large amounts of data on blockchain, (M. S. Ali, 2017) introduced model that combine the IPFS with smart contracts to make it easier for IoT devices to share data with one another. In untrusted environments, these frameworks facilitate IoT communication as well as data transfer.

3 Research Methodology

The patient records are being converted from handwritten registers to computerised data that will be stored on a cloud server and will be accessible via mobile devices from any location. Due to this mobile computing, patients are no longer required to visit hospitals, as their

mobiles will detect their body vitals and pass that information to a cloud server, which doctors or medical professionals will then access to prescribe medication.

3.1 Proposed Model

The pharmaceutical and medical industries spend a substantial sum on patient records storage. In the existing research, all medical patient records are stored on a server with limited memory that uses IPFS. When all the files are maintained in plain format, then the server will run out of memory. Conventional lossy storage methods are unable to store health data because the data must be fully recovered during decompression. This work presents a novel lossless compression technique that stores medical data quickly and efficiently. It utilises existing framework for the sharing of EHRs which integrates blockchain and the decentralised interplanetary file system on mobile cloud platform. In order to conserve memory, all medical data need to be compressed utilising data compression techniques. It is essential to compress medical data prior to storage in order to conserve storage space. The proposed model utilises LZW (Lempel–Ziv–Welch) data compression technique. The below diagram represents the proposed methodology.

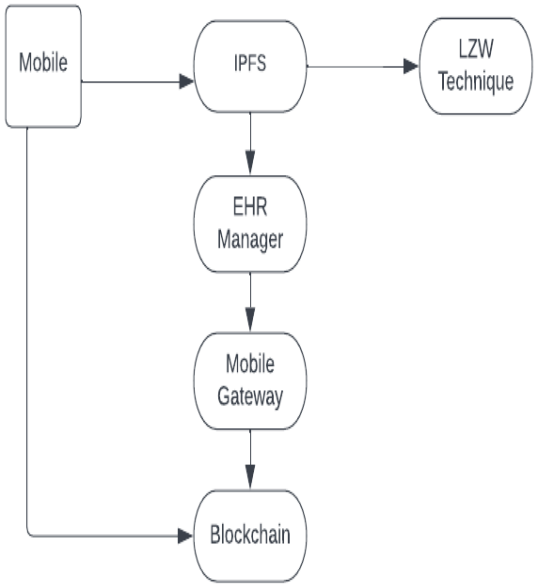

Fig 1: Proposed Methodology

LZW is a technique for lossless data compression that is quick and efficient for storing medical data. It is a dictionary-based lossless compression technique that utilises data redundancy and repetition to compress the data. When necessary, the compressed data can be recovered very quickly. LZW can be utilised in cloud environments to compress data during storage, thereby reducing data transfer and storage times. Because compression as well as decompression depend on the compiler, and as technology advances, more powerful processors become available, the compiler's speed will be increased. Because decompression

is predefined with ASCII values, the implementation leads us to the conclusion that LZW has no overhead for sending the key.

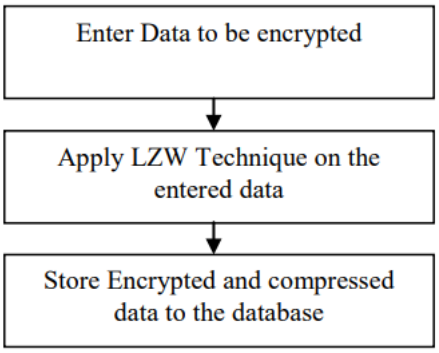

Fig 2 : Data Compression Process

4 Design Specification

In this section, architecture of proposed system will be described and introduce the concepts of data uploading as well as data sharing. The principal components of the cloud based blockchain network are outlined below. The architecture below illustrates the data flow of the model.

1. The **EHR manager** contributes significantly to the blockhain-IPFS data-sharing framework. It manages all user interactions on blockchain network, including data storage for mobile gateways and access to user data via mobile devices.
2. By adding, modifying, and removing permissions, **Admin** facilitates cloud-based financial and operational transactions and procedures. Only the Administrator, who is in charge of deploying Smart Contracts, can make changes to their underlying policies.

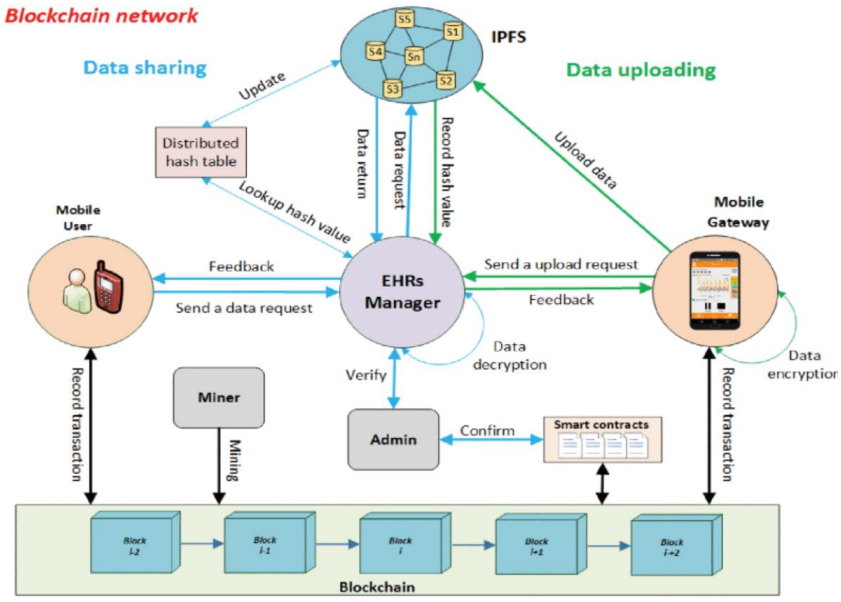

Fig 3 : Cloud Blockchain Architecture

3. **Smart contracts** describe all permitted access control system operations. The contract address as well as Application Binary Interface allow users to interact with smart contracts. By triggering transactions or messages, smart contracts are able to identify, validate, as well as grant access privileges for medical users. The smart contract with its operations are available to all entities on the blockchain. It is essential software for e-health platform.
4. **Decentralized storage** is not suitable for sharing and storing large amounts of data on blockchain. As a result, the decentralised InterPlanetary File System which is peer-to-peer file system is used, which is a promising solution for building a file-sharing platform inside the blockchain network. With IPFS, users don't need to rely on a single server to store their data; instead, they can keep their medical records in a distributed network of replication nodes with a similar file structure, which has several advantages over traditional cloud storage. IPFS facilitates file identification and access by relying on cryptographic hashes of their contents.
5. The hashcode obtained via IPFS will be saved to the blockchain. The IPFS server's memory will be quickly exhausted if all data is stored in plain text. In order to avoid having to delete data because space will be ran out, data will be compressed and kept that version around. This is made possible through data compression techniques, which reduce the amount of space required to store and transmit data while maintaining the same amount of information. The massive quantity of data typically necessitates the use of extensive computational resources. This research utilizes **LZW algorithm** which is a lossless compression technique.

   In the context of electronic health records (EHRs), health data sharing is encrypted as well as maintained in IPFS nodes, while the EHRs manager documents and stores hash values in DTH. Additionally, smart contract is integrated with IPFS to enhance decentralised cloud storage as well as managed data sharing for improved user access management. The IPFS platform-based EHR storage is configured with the a network of nodes S=S1, S2,...,Sn. As shown in the below figure, each node stores only the E-health records of patients in a specific region.

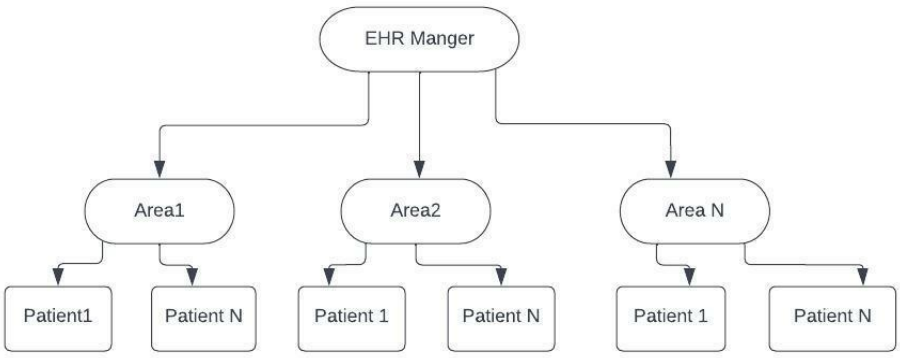

Fig 4: EHRs Storage System

The transactions within block are structured using a Merkle tree, where each leaf node represent a mobile user's data access transaction. A mobile user must give patient data (Patient ID, Area ID) for building a transaction which is signed with the user's private key at specific time in order to submit a data request (timestamp). This digital signature is meant to establish trust between the cloud server and the user.

Each block header includes a timestamp, version number, the hash utilized in the preceding block, the Merkle root hash, the target hash and the nonce. Below diagram represents the block structure.

1. **Merkle Root** is block inside a blockchain network is hash of all transaction hashes in that block. Therefore, putting the Merkle root in the block header renders the transaction tamper-resistant and can result in disc space savings.
2. A **nonce** is a created random or with semi-random number with a specific purpose. It concerns encrypted communication and information technologies. This phrase is usually known as cryptographic nonce and stands for number used once or number once.
3. The primary role of a **timestamp** is to identify the absolute instant at which a block was mined as well as authenticated by the network. A timestamp is a tiny serial number included in each block.

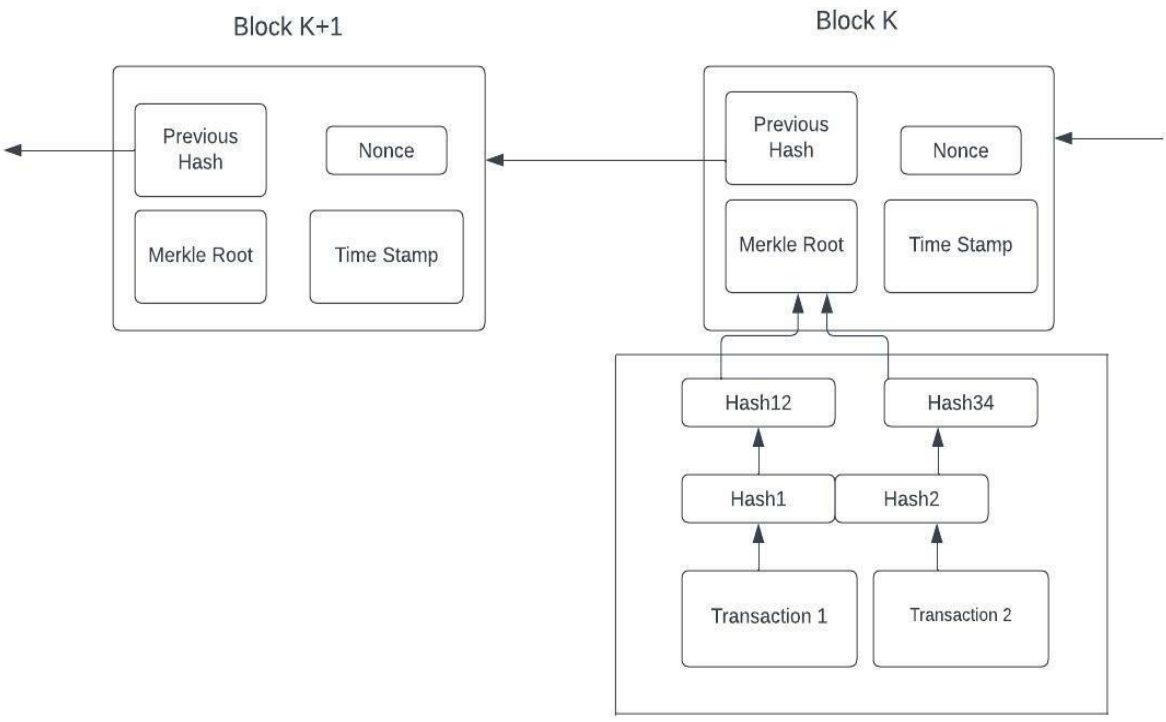

Fig 5: Data Block

4. The **previous block hash** field is contained within the block header that influences the hash of the current block. Any modification to the parent node causes the parent's hash to change. The modification of the parent's hash implies a modification of the child's "previous block hash" reference. This alters the hash of the child, necessitating

a change inside the pointer of a grandchild, which alters their grandchild. This cascade effect assures that after a block has spawned multiple generations, it cannot be modified without requiring a correction of all future blocks.

The LZW algorithm examines the data stream and applies coding rules during encoding. The algorithm transmits the encoded string if it is shorter than the dictionary's longest word, and it redefines the dictionary in opposite direction during decoding. In the world of LZW, there are two types to learn about: static and dynamic. Encoding and decoding have no effect on the dictionary in static dictionary coding. A unique code is given to each pattern that fits this description. Thereafter, the reason's corresponding code (typically picked to be smaller than the pattern) is substituted for any new occurrence of the reason, resulting in significant space savings. After compiling all of these patterns and codes, a dictionary will be there.

Smartphones and other mobile gateways need blockchain accounts and queries as transactions in order to upload data to the distributed ledger. The mobile gateway talks to the blockchain in the cloud through the blockchain client modules. When a EHRs administrator receives a transaction out from gateway, they'll send a signal to the smart contract, indicating that a fresh data upload request has arrived. By adding metadata to the pool of unsigned transactions, uploading transactions also serves as a means of archiving electronic health records inside the cloud. A user's transaction is obtained and reviewed by the EHR system administrator. The EHR administrator then notifies smart contract to verify the change. In the same way that data is uploaded, blocks of related transactions will be sent to the transaction pool and encoded by the network of miners.

5 Implementation

Step-1: Smart Contract Development
Smart contracts are designed to create a model for regulating access. In addition, an access protocol is designed which represent the work process of EHR sharing model

createAccessAccount(): With this function, an administrator can be added to the contract. When an administrator requests access to the contract, their IPFS hash is used to verify their identity and assign them the appropriate permissions. A database that includes administrative data is stored in the cloud.

createDataUserAccount(): Using this function, admin can incorporate a new user into the smart contract. When a user requests access to the contract, their ipfs hash is used to verify their identity and assign them the appropriate permissions. In addition to the rest of the system database, user data is stored in the cloud.

getDataUserAccount(): The electronic health record manager can then retrieve the data from the cloud. The smart contract requires the patient's address (including the Patient ID as well as Area ID) to be provided by a network participant. The contract then confirms and transmits a message to the EHRs manager requesting the desired data. It is the responsibility

of the EHRs manager to notify the smart contract whenever an unauthorised request is made to the EHRs system, at which point the smart contract will levy a penalty against the user making the request. The unauthorised mobile entity is penalised with a warning message. Below is the code snippet for the above mentioned functions.

```
public class SmartContract extends Contract {
    private static final String BINARY = "60806040523480156100105760080fd5b506104218061002060000396000f3fe

    public static final String FUNC_GETACCESSACCOUNT = "getAccessAccount";

    public static final String FUNC_CREATEACCESSACCOUNT = "createAccessAccount";

    public static final String FUNC_GETDATAUSERACCOUNT = "getDataUserAccount";

    public static final String FUNC_CREATEDATAUSERACCOUNT = "createDataUserAccount";

    protected SmartContract(String contractAddress, Web3j web3j, Credentials credentials, BigInteger gasPr
        super(BINARY, contractAddress, web3j, credentials, gasPrice, gasLimit);
    }

    protected SmartContract(String contractAddress, Web3j web3j, TransactionManager transactionManager, Bi
        super(BINARY, contractAddress, web3j, transactionManager, gasPrice, gasLimit);
    }

    public RemoteCall<String> getAccessAccount() {
        final Function function = new Function(FUNC_GETACCESSACCOUNT,
                Arrays.<Type>asList(),
                Arrays.<TypeReference<?>>asList(new TypeReference<Utf8String>() {}));
        return executeRemoteCallSingleValueReturn(function, String.class);
    }
```

Fig 6: Smart Contract

```
public static String createAccessAccount(String pid,String ipfs_hash){
    String result = "none";
    try{
        setup();
        System.out.println("Address "+address);
        SmartContract sc = SmartContract.load(address, web3j, credentials, ManagedTransaction.GAS_PRICE, C
        String access = sc.getAccessAccount().send();
        if(access.length() > 0)
            access = access+"#"+pid+","+ipfs_hash;
        else
            access = pid+","+ipfs_hash;
        sc.createAccessAccount(access).send();
        result = "success";
    }catch(Exception e){
        e.printStackTrace();
    }
    return result;
}
```

Fig 7: Code Snippet for AccessSmartContract

Step-2: LZW Compression

The below method collects the name,date of birth,address and condition details from the patient and then performs the LZW compression on the data.Compression with LZW works by reading a series of symbols, then compressing that string, and finally decoding the code. Code tables are used in LZW compression, as well as a popular pattern for number of entries in the table is 4096. Bytes from input file(Patient data) are always represented by the corresponding codes 0-255 inside the code table. When encoding is started, the code table

maintains initial 256 entries in it. The rest of the table is blank. By representing bytes with codes between 256 and 4095, the data will be compressed.

```
def PatientData():
    if request.method == 'POST':
        global pid
        global api
        pid = pid + 1
        name = str(request.form['t1']).strip()
        date = request.form['t2']
        address = request.form['t3']
        phone = request.form['t4']
        condition = request.form['t5']
        data = str(pid)+","+name+","+date+","+address+","+phone+","+condition
        original_size = sys.getsizeof(data)
        compressed = zlib.compress(data.encode())
        compress_size = sys.getsizeof(compressed)
        f = open(str(pid)+".txt", "w")
        f.write(data)
        f.close()
```

Fig 8: LZW Algorithm

Step-3: Data Encrytion

The following step is an encryption process where the patient id is used with an encryption algorithm to further secure the compressed file. The EHRs manager checks the request IDs of authorised access and it looks up for hash value of data file which contains the request data in the distributed hash table in order to retrieve the necessary data from IPFS storage. The IPFS system returns the encrypted file after data uploading. The EHRs administrator decrypts the data file using an asymmetric encryption method and its private key, and then sends the decrypted data back to the requesting party. The gateway's blockchain client module uploads the encrypted file to an IPFS cloud storage server.

```
new_file = api.add(str(pid)+".txt") #adding encrypted model to IPFS
hashcode = new_file['Hash']
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('localhost', 3333))
hashdata = 'createrecord,'+str(pid)+","+hashcode
message = client.send(hashdata.encode())
data = client.recv(100)
data = data.decode()
print(data)
os.remove(str(pid)+".txt")
color = '<font size="" color="black">'
output = color+" Record stored in IPFS and in Blockchain with ID : "+str(pid)+" <br/>Hashcode = "+
output += "Original Data Size : "+str(original_size)+"<br/>Compressed Data Size : "+str(compress_s
height = [original_size, compress_size]
bars = ('Original Data Size', 'Compress Data Size')
y_pos = np.arange(len(bars))
plt.bar(y_pos, height)
plt.xticks(y_pos, bars)
plt.title("Normal & Compressed Data Size  Graph")
plt.show()
return render_template("Patients.html",error=output)
```

Fig 9: Coding scnippet for storing data in IPFS

For mining purposes, miners will collect the transactions into blocks and add them to the transaction pool. All confirmed transactions will be added to the distributed ledger (blockchain) and made available to all participants. The user can modify their transaction details through the blockchain client built into their gateway. An access protocol is designed that is carried out whenever a user performs a transaction inorder to request EHRs on cloud,

allowing to function the user(Patient) access control mechanism for Electronic health record sharing.

Access Control System:

A new user-initiated transaction has been received by the EHRs manager. The Electronic health record manager will inquire about the requesting patient's id and submit it to contract for verification. EHRs manager sends a transaction including a user-patient id. The administrator will look up the requester patient ID in the smart contract's policy list to see if it has access to the requested resources. If the patient's identifier is present in the database, the request is granted and the user is given access to the data. If this is not done, the smart contract would then issue penalty for this action.

Once the contract is given permission to process the new transaction, it will decipher it. The administrator can now access the transaction's Area ID as well as Patient ID from within EHRs. The administrator then communicates with the EHRs manager, asking them to retrieve necessary patient data from the distributed cloud storage system. Once the desired EHRs have been located, the Electronic health record manager will transmit them to the requesting party using a special off-chain network designed for cloud-mobile user interaction. With the data processing now complete, a new transaction has been added to the blockchain and shared with all nodes in the network.

When analysis of access controls allows retrieval of EHRs along with identification of threats to cloud-based data storage, the process is considered to be complete. The ability of the access control approach to assess and prevent attackers to the electronic health system is crucial to the success of the blockchain mission to enable secure EHRs sharing.

6 Evaluation

To start Ethereum tool, start eth.bat file is executed. The smart contract is deployed to ethereum tool and then IPFS server is started. A user on the go, like a doctor who needs access to his patients' electronic health records in the cloud, an Ethereum account will be created as well as user information will be registered in order to interact with the blockchain via an app with a user interface. The smart contract would authenticate as well as detect unauthorised access using the access protocol and a set of predefined policies. The requester will be warned and then delete their information from the EHR database if it turns out that they have made such a request. In the case of unauthorised access, the smart contract will also produce a matching transaction. It is evident that blockchain can be utilized effectively to address issues outlined in the related work that specifies controlling patient information and tracking user access, thereby improving system reliability and data privacy.

Fig 10 : Screen for index page

After the cloud EHRs manager approves his request, he begins the transaction to access EHRs by entering the patient's address (includes PatientID and AreaID). The results of his data access query will then be returned by EHRs system, and his mobile interface will also be updated to reflect these changes. In this way, the doctor is able to obtain the patient's complete medical history for the purposes of conducting an in-depth evaluation and providing the best possible care.

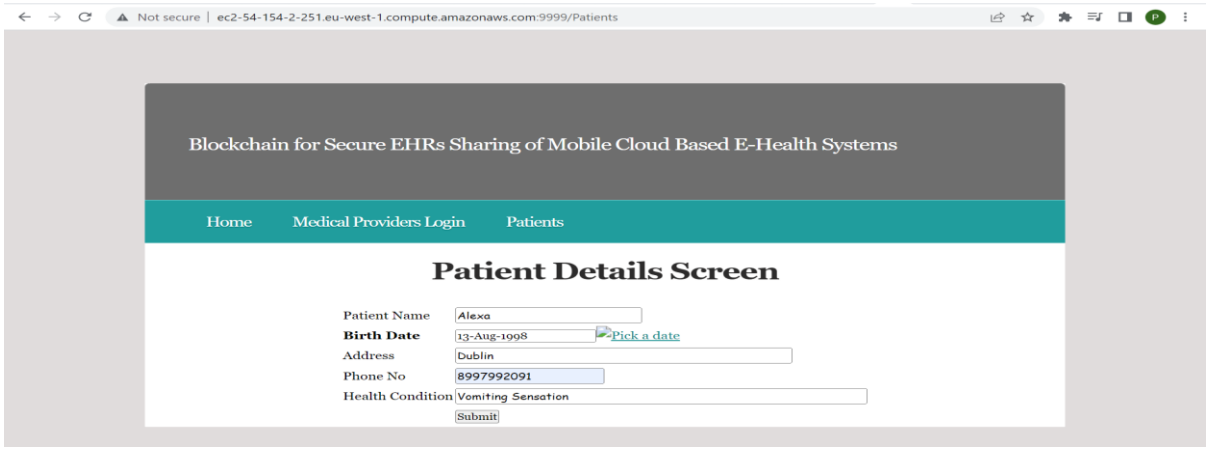

Fig 11: Patient detail screen

This concludes the process of gaining access to electronic health records, and the cloud miner will now append this transaction to the blockchain and make it available to all participants in the network. As a result, patients can verify the identities of those accessing their electronic health records (EHRs), ensuring that each user has permission to access their information while also protecting the integrity of the network.

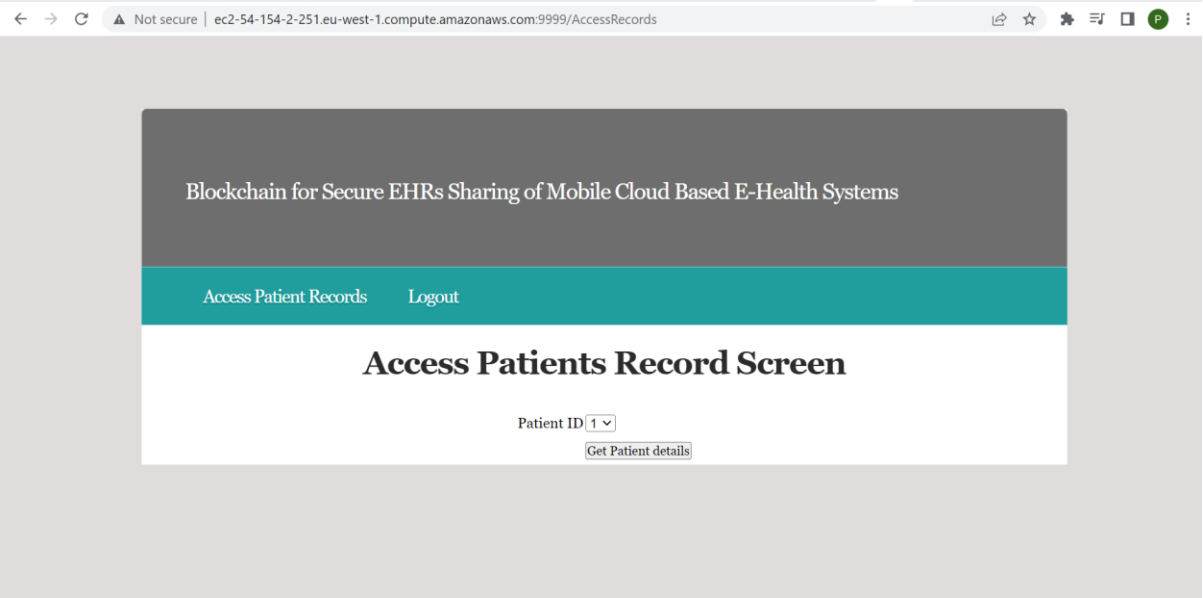

Fig 12: Access Patients Record Screen

The application is then deployed to AWS using EC2 instance. It takes more time for the smart contract-based authentication mechanism to process user requests than the non-authenticated scheme. Time spent verifying users' identities and granting them access is the source of this delay. The worst-case scenario (seven requests) still results in an additional overhead of around 100ms, which is acceptable in most situations. This finding exemplifies proposed model lightweight authentication and authorization framework. The average time it takes for a cloud service to handle a large number of requests for access is measured in the below figure.

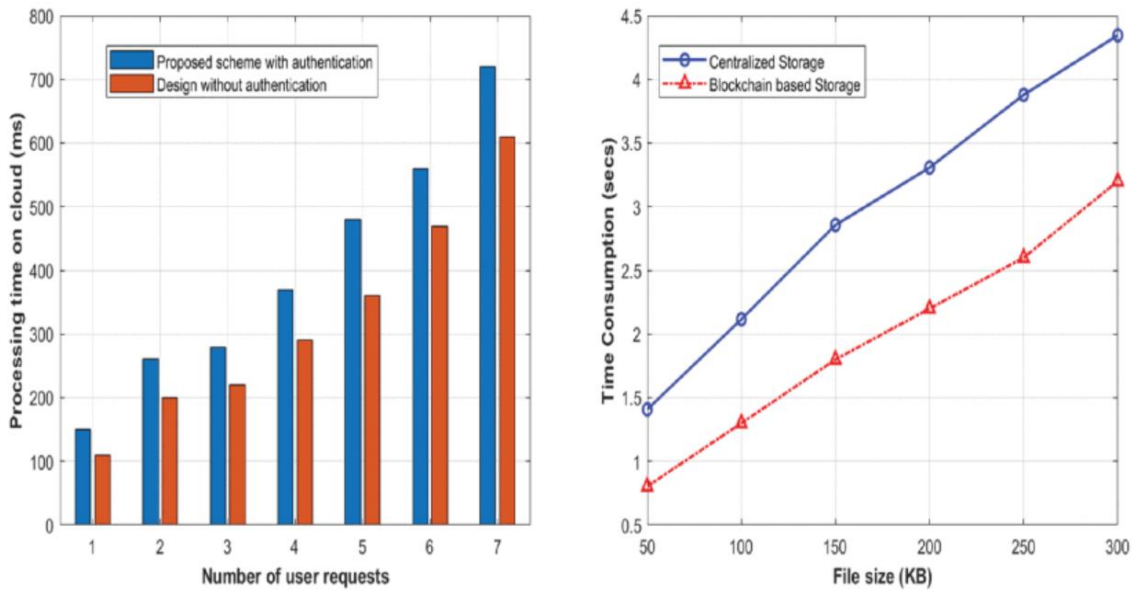

Fig 13: Processing time and time consumption graphs

Compressing data is a useful technique for reducing its size of textual data, storing the very same amount of data in relatively fewer bits, and thereby decreasing the need for storage space, resources, or transmission capacity. By reducing the size of the data, retrieval times can be reduced, leading to a more efficient process. It helps make the most of hard drives and bandwidth. This work allows the user to precisely balance the costs of storage, computation, and bandwidth.The graphical representation for the original data and compressed data is presented in the below figure. The compressed data occupies less storage in IPFS than the original data size.

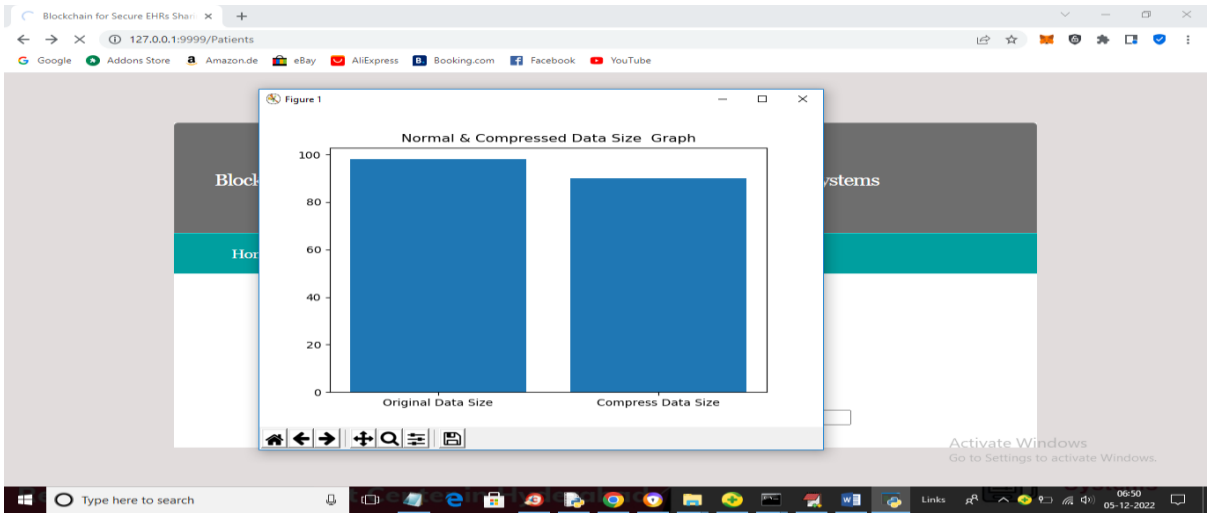

Fig 14: Original Data Vs Compressed Data

6.1 Experiment / Case Study 1

In case study 1, when the patient registration has been completed on click of submit, one can examine the original data size as well as the compressed data size. The compressed data is less when compared to the original and thereby success is observed by occupying less storage after compression.

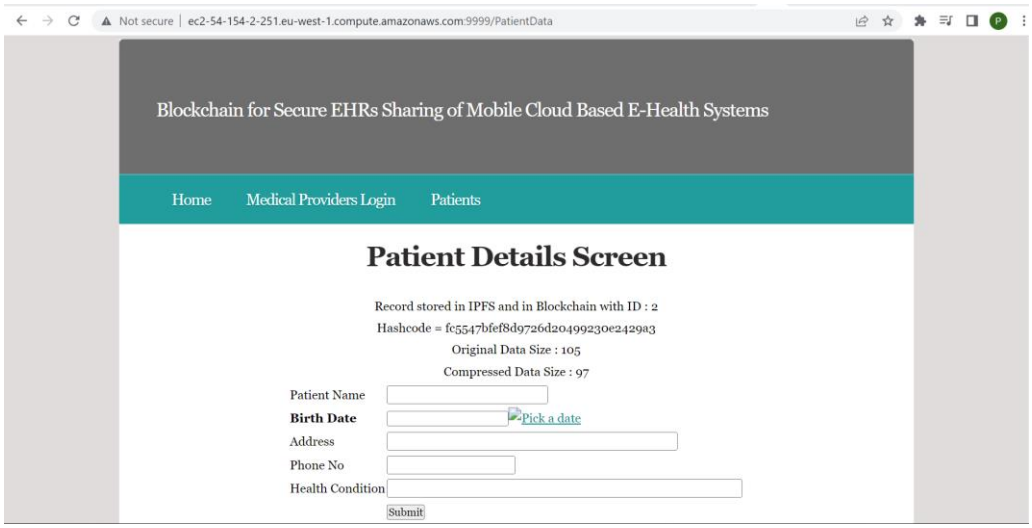

Fig 15: Screen for Original and Compression Data Size

6.2 Experiment / Case Study 2

In the case study 2, access is restricted to electronic health record (EHR) resources in the cloud except to authorised entities. Each request will be verified by EHR manager and then provided access to those entities. Here success can be observed by restricting access to unauthorised entities.

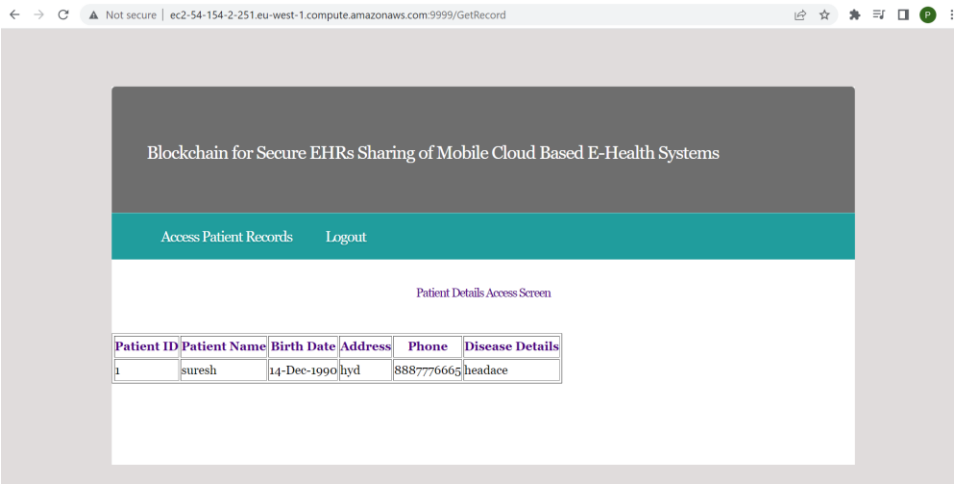

Fig 16: EHR access to authorised entities

6.3 Discussion

The access control scheme protects individual's privacy and ownership of their data by leveraging the immutable ledger and automated processes of smart contracts. The cloud storage used has a built-in firewall, based on the identities of its users and the authorization provided by a smart contract. Additionally, blockchain network's consensus mechanism will invalidate illegal transactions and remove them from the ledger. The distributed nature of the blockchain means that all participants can keep an eye on every transaction and message without exception. As a result, users can easily keep an eye out for any tampering with cloud medical records and report it to the cloud manager, ensuring that all sensitive information remains secure. The proposed system is protected from outside interference with a thorough investigation of the various facets of its design. The adaptability, availability, high system integrity, and data privacy that come with proposed design are additional attractive features that have potential utility in healthcare contexts.

Uploading patient records to the cloud raises important privacy concerns for healthcare blockchain that have been largely ignored by previous research. For instance, a prying cloud server in a blockchain network hosted in the cloud may be able to access private patient information by snooping in databases that were intended for other purposes. Although the blockchain facilitates the tracking of transaction records by various entities, malicious miners may infer personal information about users, such as their location as well as their usage patterns, during the mining process. Inventive strategies that maximise both upload efficiency and user safety may prove useful in such situations. The use of reinforcement learning (RL) uploading techniques has recently emerged as a potentially game-changing approach to addressing these pressing healthcare issues (Min M. W., 2018).

Traditional cloud-based health services may not be suitable for time-sensitive health applications due to high network latency caused by the physical distance to mobile platforms. In order to overcome these obstacles, mobile edge cloud may be a viable option for providing health service to mobile users with minimal network latency. To optimise data processing as well as transaction communication for lower latency, the healthcare industry requires solutions for lightweight type blockchain design.

7 Conclusion and Future Work

This paper presents a new method of sharing electronic health records (EHRs) made possible by blockchain and mobile cloud computing. Problems with current EHR sharing systems are identified, and effective solutions are presented through the use of a working prototype. The goal of this work is to integrate the existing blockchain IPFS framework with data compression methods. Furthermore, it is concerned with the development of a reliable access control mechanism that uses a single smart contract for handling user access for the purpose of ensuring the efficient and secure exchange of electronic health records. Health entities can communicate with the Electronic health record sharing system through a built-in mobile Android app, and its performance is being studied by deploying the application to the Amazon cloud. The results of proposed work implementation demonstrate its ability to facilitate the secure and rapid exchange of medical data by users in mobile cloud environments, in comparison to more traditional schemes. You can see the effects of the implementation in the storage graphs of both the compressed data and the original data. To ensure patients confidentiality and the safety of their data, an access control system has been developed that can detect and block intrusions into the e-health system. In addition to highlighting the benefits of the proposed work over pre-existing solutions, this work provides a security analysis and in-depth evaluations of various technical features of the suggested system.

Combining LZW as well as run length encoding into a single algorithm for data compression is a promising direction for future research. When compared to other algorithms, this hybrid method achieves a higher compression ratio while simultaneously reducing the amount of time spent on compression and decompression.

References

References should be formatted using APA or Harvard style as detailed in NCI Library Referencing Guide available at https://libguides.ncirl.ie/referencing
You can use a reference management system such as Zotero or Mendeley to cite in MS Word.

Beloglazov, A. and Buyya, R. (2015). Openstack neat: a framework for dynamic and energy-efficient consolidation of virtual machines in openstack clouds, *Concurrency and Computation: Practice and Experience* 27(5): 1310–1333.

Feng, G. and Buyya, R. (2016). Maximum revenue-oriented resource allocation in cloud, *IJGUC* 7(1): 12–21.

Gomes, D. G., Calheiros, R. N. and Tolosana-Calasanz, R. (2015). Introduction to the special issue on cloud computing: Recent developments and challenging issues, *Computers & Electrical Engineering* 42: 31–32.

Kune, R., Konugurthi, P., Agarwal, A., Rao, C. R. and Buyya, R. (2016). The anatomy of big data computing, *Softw., Pract. Exper.* 46(1): 79–105.

R. Wu, G.-J. Ahn and H. Hu, "Secure sharing of electronic health records in clouds", *Proc. 8th Int. Conf. Collaborative Comput. Netw. Appl. Worksharing (CollaborateCom)*, pp. 711-718, Oct. 2012.

A. Ibrahim, B. Mahmood and M. Singhal, "A secure framework for sharing electronic health records over clouds", *Proc. IEEE Serious Games Appl. Health*, pp. 1-8, May 2016.

Z. Ying, L. Wei, Q. Li, X. Liu and J. Cui, "A lightweight policy preserving EHR sharing scheme in the cloud", *IEEE Access*, vol. 6, pp. 53698-53708, 2018.

V. Ramani, T. Kumar, A. Bracken, M. Liyanage and M. Ylianttila, "Secure and efficient data accessibility in blockchain based healthcare systems", *Proc. GLOBECOM*, pp. 206-212, Dec. 2018.

N. Rifi, E. Rachkidi, N. Agoulmine and N. C. Taher, "Towards using blockchain technology for eHealth data access management", *Proc. IEEE 4th Int. Conf. Adv. Biomed. Eng.*, pp. 1-4, Oct. 2017.

Q. I. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du and M. Guizani, "MeDShare: Trust-less medical data sharing among cloud service providers via blockchain", *IEEE Access*, vol. 5, pp. 14757-14767, 2017.

X. Liang, J. Zhao, S. Shetty, J. Liu and D. Li, "Integrating blockchain for data sharing and collaboration in mobile healthcare applications", *Proc. IEEE 28th Annu. Int. Symp. Pers. Indoor Mobile Radio Commu. (PIMRC)*, pp. 1-5, Oct. 2017.

S. Wang, Y. Zhang and Y. Zhang, "A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems", *IEEE Access*, vol. 6, pp. 38437-38450, Jun. 2018.

M. Steichen, R. Norvill, B. F. Pontiveros and W. Shbair, "Blockchain-based decentralized access control for IPFS", *Proc. IEEE Blockchain*, pp. 1499-1506, Jul. 2018.

Y. Chen, H. Li, K. Li and J. Zhang, "An improved P2P file system scheme based on IPFS and blockchain", *Proc. IEEE Big Data (Big Data)*, pp. 2652-2657, Dec. 2017.

M. S. Ali, K. Dolui and F. Antonelli, "IoT data privacy via blockchains and IPFS", Proc. 7th Int. Conf. Internet Things, pp. 14, Oct. 2017

Du, Y. and Yu, H., 2020, August. Medical Data Compression and Sharing Technology Based on Blockchain. In International Conference on Algorithmic Applications in Management (pp. 581-592). Springer, Cham.

Akrasi-Mensah, N.K., Tchao, E.T., Sikora, A., Agbemenu, A.S., Nunoo-Mensah, H., Ahmed, A.R., Welte, D. and Keelson, E., 2022. An Overview of Technologies for Improving Storage Efficiency in Blockchain-Based IIoT Applications. Electronics, 11(16), p.2513.

Min, M., Wan, X., Xiao, L., Chen, Y., Xia, M., Wu, D. and Dai, H., 2018. Learning-based privacy-aware offloading for healthcare IoT with energy harvesting. IEEE Internet of Things Journal, 6(3), pp.4307-4316.

Alharby, M., Aldweesh, A. and Van Moorsel, A., 2018, November. Blockchain-based smart contracts: A systematic mapping study of academic research (2018). In 2018 International Conference on Cloud Computing, Big Data and Blockchain (ICCBB) (pp. 1-6). IEEE.